DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Seminar paper

Data Analytics for Cybercrime and undesirable online behaviour

# Polarity mining of Stack Overflow answers in the C and Python subcommunities

**Luca Mario Hohmann**

**Supervisor: Prof. Dr. Jens Großklags**

I confirm that this paper is my own work and I have documented all sources and material used.

15.03.2020 Luca Hol

Munich, Submission date Luca Mario Hohmann

# Abstract

The question and answer forum Stack Overflow gained the stereotype of being an unfriendly and unwelcoming place on web. This paper uses sentiment analysis with the VADER sentiment tool to analyze answers on Stack Overflow. The goal is to be able to make a statement about the answering cultures in two subcommunities of the website, namely the C and the Python community. First this paper explains the used methodology to gather the data, then the used tool is explained. Afterwards it evaluates 3 questions at the level of a whole subcommunity. First it looks at the over all polarity of all answers. Then it compares the two subcommunities against each other. Next it analyzes the trend of polarity over the years since the launch of the website. Afterwards 2 questions at the user level, including user controlled features are proposed and answered. First the editing feature for answers is investigated and analyzed whether it promotes different interaction of users and a different sentiment of the answers. Following this question, the top five contributors of both communities are investigated in order to analyze whether the reputation system - with it's benefits for experienced users - promotes better and friendlier answers compared to the average user. In the end, the results of the sentiment analysis tool for Software Engineering texts are discussed and put into perspective. The results of the four questions indicate that the polarity of answers on Stack Overflow decreased, while the number of users increased over time for these two communities. However the overall polarity is still positive and not overwhelmingly negative as it is portrayed by the stereotype. The feature of editing answers seems to improve the over all quality and positive interaction rate of answers significantly in the Python community. The top users of both communities do not show exceptionally positive answers but most of the time a polarity below the community average. Nevertheless in recent years and with growing experience, the top users increase their polarity above the community average in the Python community. This whole analysis is for many parts at least partially based upon the results of the sentiment tool VADER. The tool shows different behaviours when inputting the same texts in other formats which may lead to completely opposite interpretations of texts. When comparing the results to a manual rating of texts one of the possible values is very close to it but the other value, which is the value used throughout the paper, shows significant other classifications than the manual rating.

# Contents

# 1 Introduction

The Stack Exchange network gained over 9 billion views and reached over 100 million users in the year 2018. Stack Overflow gained over 2.5 million answers and 2 million questions in that year [1]. These statistics make Stack Overflow one of the most important community based question and answer forums (CQ&A) for software engineering on the web. Software Engineers of all experience levels visit Stack Overflow to find answers to their own questions and answer questions of other users [2]. The website embeds a reputation system which rewards active user that produce high quality content. Every post in form of a question, an answer or a comment generates reputation for the user. Interactions of other users such as upvoting a question or answer and accepting an answer as solution to a questions generate additional reputation. This system should encourage friendly, open and high quality questions and answers.

Such amounts of users and diversity will eventually lead to problems on a platform like Stack Overflow. One of the main problems of Stack Overflow is the stereotype as having a very unfriendly and unwelcoming answering culture towards "[...] newer coders, women, people of color, and others in marginalized groups" [3]. Over time the Stack Exchange team started different campaigns to analyze and address the problem [3] [4]. Projects like the "Stack Overflow Comment Evaluator 5000" [4] focused on the polarity of 3992 comments and answers. They were rated manually by 57 persons in 2018. These roughly 4000 posts are less than one percent of the answers of that year. Manual rating can not cover the number of answers Stack Overflow is gaining every year and additionally cover the years since the start of the website in 2008. Therefore this seminar paper analyzes the usage of automated sentiment analysis with "Valence Aware Dictionary and sEntiment Reasoner" (VADER) [5] for the polarity rating of Stack Overflow answers. Hence the amount of answers at Stack Overflow, this analysis focuses on the Python and C sub communities within Stack Overflow. These two communities account for nearly 9.3% of all answers on the website. The goal is to compare these two according to their answering and communication culture and answer whether these communities are as negative as the stereotype of Stack Overflow. Additionally the usability of VADER for Software Engineering texts is examined.

First the paper describes the data acquisition and processing. Then it describes the VADER sentiment analysis tool. Next it examines the question whether the C and Python communities have a negative answering culture and whether VADER can be utilized for further research in this area. Afterwards the results are discussed on their statistical validity as well as the behaviour of VADER for software engineering texts. In the end a short outlook with suggestions for future research follows.

# 2 Data Acquisition and Processing

## 2.1 Data Structures

The Stack Exchange Network provides free yearly data dumps for every forum at the Online Archive [6] as well as a live database for Stack Overflow, which is updated once a week [7]. The data dumps consist of one XML-file for each category of data they provide. The live database returns the tuples in a browser view and allows a download as CSV file. The XML files contain everything since the start of the corresponding forum. Because the Stack Overflow file contains over 69GB of data, but only a few percent are used for the analysis, I decided to use the live database. The live database provides all properties of posts to create SQL queries. These queries are rate limited at 50.000 returning tuples. A query finding more tuples displays the first 50.000 tuples and cuts of every additional tuple. Therefore the queries need to be constructed to stay under this limit. The aim of the paper is to analyze all answers since 2008. The number of answers in the C and Python community reached 2.704.067 items until the 2nd of December 2019. In order to download the data, I created SQL queries with different time ranges. The C subcommunity required quarterly queries as the number of answers start to drop since 2014 (see Figure 2.1). In contrast the Python community grew so significantly over time (see Figure 2.2) that queries for single months were required [8]. Downloading all data points is a trial and error process in which I had to test the number of months that exceed the rate limit [8]. This approach results in 2.36GB of data. After the download of the CSV files, I created a script to join the monthly files into yearly files for C and Python.

## 2.2 Data Preprocessing

Then the preprocessing of the questions began. In order to mislead the sentiment analysis tool as little as possible, every syntactical part as well as all source code should be removed from the answers. Stack Exchange uses HTML tags to display different texts in the answer's bodies. The most important tags are the <*p*>, <*code*>, <*pre*>, <*a*> and <*blockquote*> tags. The *p* tag marks new paragraphs and is often used when a the previous paragraph ends or a blockquote or code segment ends. Every other HTML tag, which is used on the site, may occur within the paragraph tag. The *code* and *pre* tag often occur as a pair in which the *pre* tag embeds the *code* tag. Inside the *code* tag, a user can write source code which is displayed with a different background color, different font properties and syntax highlighting. Links are embedded through the <*a*> tag. The *blockquote* tag often contains quotes from other user's texts and source code. The text of a *blockquote* displays text with a different background and

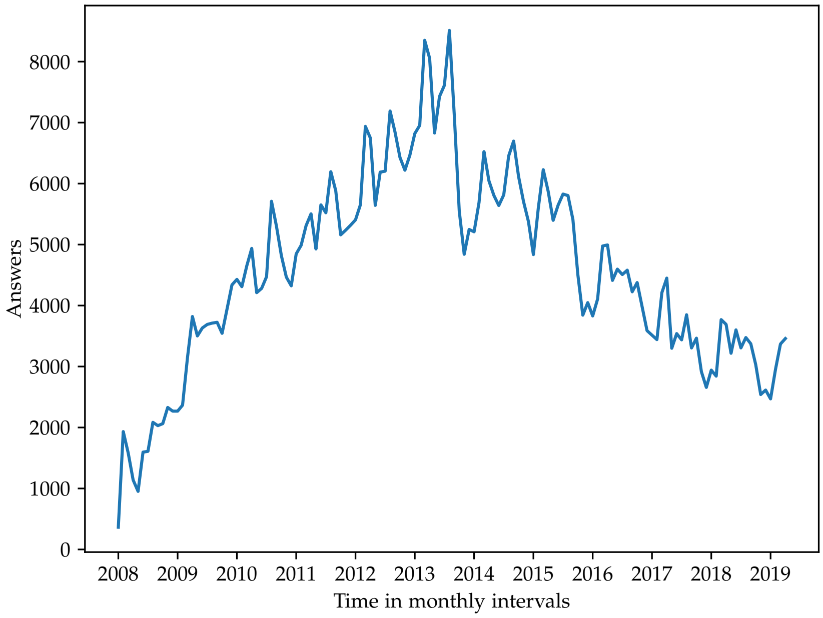

Figure 2.1: Monthly numbers of answers by the C community [August 2008, November 2019].

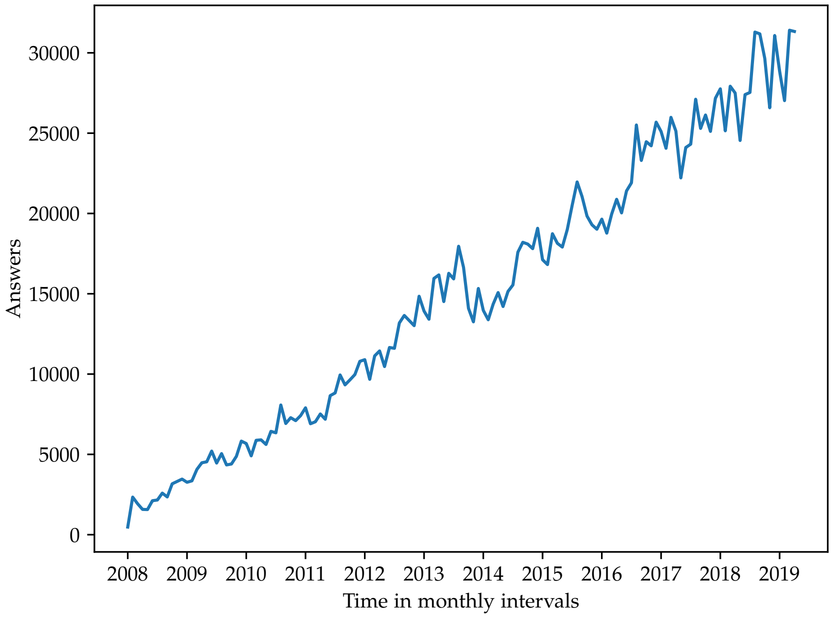

Figure 2.2: Monthly numbers of answers by the Python community [August 2008, November 2019].

can contain all other tags as well.

In order to remove the tags, two different approaches are required. HTML Tags that embed normal text only need to be removed without affecting the body. These were removed by scanning the text for the opening symbol of a tag < and then skipping everything until the closing symbol > is found. Every starting tag also has a closing tag, which is treated like a new opening tag, because the HTML code within the tags is not tested for specific symbols. On the other hand the *link* and *code* tag also require the deletion of the embedded text. The reason is that code may disturb the sentiment analysis, because of unusual symbols like braces, semicolons as well as programming related phrases that might be viewed as a negative or positive phrase outside of software engineering. Links can also contain words that might distort the results as users can link to arbitrary sites with any kind of URL. These might contain very provocative titles to lure people towards visiting the website, but can affect the result of a short text in a positive or negative way. These two tags were removed by finding the opening tag <*code*> or <*a*> and then ignoring everything until the corresponding closing tag </*code*> or </*a*> is detected. These heuristics try to minimize the noise in the answers themselves. However some answers were blank after processing them, because they only contained a link or source code. From the original number of answers, 45.702 answers were removed, because they did not contain any text after the cleaning process. This process results in 2.658.365 data points which are stored inside a PostgreSQL database with the corresponding results of the sentiment analysis. The database contains one table with a relation that is structured as seen in Table 2.1.

Before inserting the answers into the database, each answers needs to be evaluated by VADER.

Table 2.1: Database Relation

| attribute | usage |
|---|---|
| id (primary key) | id of the question |
| community (primary key) | 0 (C) or 1 (Python) |
| parentid | id of the question |
| creationdate | date of creation |
| score | voting score of answer |
| body | answer text |
| owneruserid | id of user answering |
| lasteditdate | date of last edit, null if not edited |
| lasteditoruserid | id of last editor, null if not edited |
| commentcount | number of comments |
| positiveaverage | average percentage of positivity over each line |
| neutralaverage | average percentage of neutrality over each line |
| negativeaverage | average percentage of negativity over each line |
| compoundaverage | average percentage of compound over each line |
| positiveall | positivity for the whole answer |
| neutralall | neutrality for the whole answer |
| negativeall | negativity for the whole answer |
| compoundall | compound for the whole answer |

# 3 VADER sentiment tool

Vader is a "lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media" [5]. It uses encoded rules as well as ground truths for analyzing texts word by word and calculating the score of a given string. The ground truth was created by multiple persons to ensure a greater consensus on the polarity of texts from tweets, newspaper as well as review snippets and other sources [5]. Vader itself is a simple python library that takes a string of characters, analyzes it and returns a dictionary of four polarity values. The positivity, neutrality and negativity values display to which degree the string falls into the three categories. A string with a result of 55% pos, 45% neu and 0% neg contains 55% positive and 45% neutral words and phrases. However the analysis also returns a compound value which indicates the total polarity of the string. It sums the polarity values of each word in the string. Then it applies the mentioned lexicographical rules of the tool to them and normalizes the result to the range of [-1, 1] [5]. The compound value is the interesting value of the four as it corresponds to the polarity, which is relevant for the analysis.

As seen in Table 2.1, the database contains every valence value twice. The reason behind this is the way the values are calculated and the way VADER allows string input. The average values are the summation of the corresponding values of each line analyzed independently. For this reason my scripts [8] split a text with seven lines into seven independent strings. These are processed by VADER, which returns seven dictionaries of values. The seven dictionaries are then summed up and divided by seven in order to receive the four average polarity values. This compound value is referenced as 'average compound'. In contrast, the values representing the whole answers are calculated by inputting the whole question's body to VADER an receiving a single dictionary. This compound value is referenced as 'all compound' in the following chapters. The result of these different approaches are discussed in section 4.3. As of writing of the paper, VADER does not allow additional input information for the evaluation. Thus the number of comments or the total voting score of each answers are not taken into account directly.

# 4 Results

In this chapter, the research questions and results of the data evaluation are presented. First the question is stated and explained. Then the relevant values, thresholds and scales are presented and classified in order to give an answer. In order to acquire the final answers, three question at the community level and two on the user level are examined. These can give a better understanding of the dynamics of Stack Overflow and show how users of the site influence the website's system. In the end the question whether VADER can be used to classify Software Engineering texts is answered.

## 4.1 Community level questions

### 4.1.1 Is the polarity of answers in the C and Python subcommunities as negative as the stereotype of Stack Overflow?

There exists the general perception that Stack Overflow is an unfriendly CQ&A forum. This perception not only holds for external people but also for the people behind Stack Overflow, as multiple campaigns on their blog showed. Because the whole of Stack Overflow is too much data to analyze in one semester, the analysis focuses on Python, which is used by nearly ever other developer, and C, which is still used by every 5th developer [2]. Therefore this perception should hold for these two communities as well, especially considering over 2.7 million answers were written since the launch of the website. In the paper that introduced VADER, a compound value greater than 0.05 can be considered as positive post and a compound value of less than -0.05 as negative [5]. According to these thresholds, a value difference greater than 0.1 can impact the interpretation of the polarity. Consequently this paper considers value changes greater than 0.1 as a significant change of polarity. This paper also looks at more extreme cases in later questions.

The data set contains almost every answer of the subcommunities, so no statistical assumptions are necessary. The average compound values (see Table 4.1) for the whole data set are above the threshold of 0.05 and can be considered positive. The standard deviation for the 'average compound' is 0.23 and 0.44 for the 'all compound' value. These standard

Table 4.1: The mean of both compound values over the whole data set (rounded to 2 digits)

| Compound (avg) | Compound (all) |
|---|---|
| 0.10 | 0.23 |

deviations display that compound values can be found in the positive as well as negative polarity range. However, the overall analysis shows a positive polarity for most of the answers.

However, the two compound values show a great discrepancy in positivity as the 'all compound' value is more than double compared to the 'average compound'. One of the statistic is either scaled up or down by a factor of two, as different values such as the standard deviation, mean or variance are often close to a scaling of two to their counterpart. The reason behind this difference are the distinct input values for the analysis as described in chapter 3. These results are discussed in section 4.3. On average every answer is positive.

### 4.1.2 Does a significant polarity difference between the C and Python community exist?

The second important question of this paper tries to compare the two sub communities. C is chosen as a representation of an older and much slower developing technology. Compared to C, which does not receive great language updates anymore, Python is under constant development, new libraries are built rapidly and is therefore chosen as representation of a modern technology language. In discussions and forum questions users refer to answers from the C or C++ community when talking about very negative examples. Meanwhile examples of very unfriendly answers from the python community are less present.
The result of the analysis in Table 4.2 shows that both communities can be considered positive as the mean values are all above 0.05. The 'all compound' values can be considered very positive. The differences between the two communities are not very large for the mean compound values and standard deviations. Only the standard deviation for the 'all compound' shows a difference of 0.07 points which results in a broader distribution of values in both the positive and negative polarity range.

Table 4.2: Mean compound value and standard deviation per community overall posts

| | C | Python |
|---|---|---|
| Compound (avg) | 0.09 | 0.10 |
| Compound (all) | 0.21 | 0.23 |
| Standard deviation (avg) | 0.24 | 0.23 |
| Standard deviation (all) | 0.50 | 0.43 |

In both communities the average over the 'all compound' value is higher than the average over the 'average compound' value. The difference is the result of the sentiment analysis process as described in chapter 3. This behaviour difference cannot be considered stable as the results are larger than the neutral value range. These huge disparities could lead to different interpretation of the polarity as one value could be less than -0.05 and the other greater than 0.05 at the same time. Such a result leaves the question which value to consider. However, for any further question on the 'compound all value' will be considered. The reasoning behind this decision is described in section 4.3.

### 4.1.3 Is a trend of rising negativity recognizable in recent years?

In recent years, the number of Stack Overflow users grew strongly as shown in Figure 4.1. For the first five years, the number of registered users grew exponentially and increases linearly since 2013. A rising number of users results in a melting pot of people from different cultural and social backgrounds [2]. With a rising number of participants, the potential problems of Stack Overflow grow as well. This effect is reflected by the number of blog posts, forum discussions and analysis over the years. One of the first blog posts aiming into the direction of unfriendly user behaviour dates back to 2012 and is titled "Stackoverflow Is A Difficult Community to Participate In" [9]. It deals with negative voting and answering behaviour of users, which the author found during his participation on the website. After this article, the number similar stories started to increment, especially in 2015 [10][11]. These indicators of people's perceptions reached a new level in 2018, when two official post on the Stack Overflow Blog were released [3] [4].

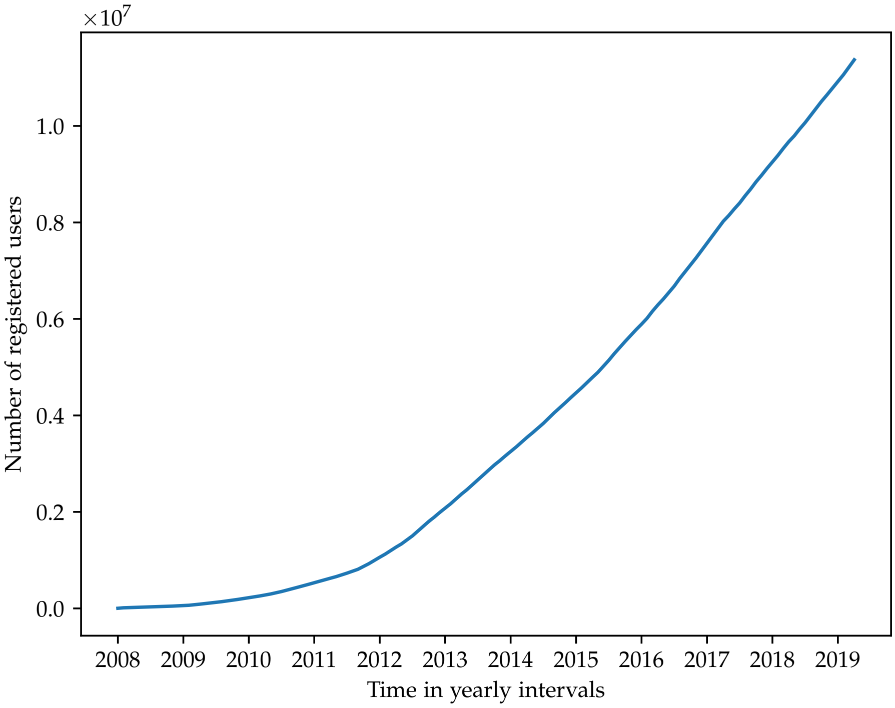

Figure 4.1: Yearly number of registered users on Stack Overflow [August 2008, November 2019]

This development of opinions and perceptions of the website lead to the question "How does the answer polarity change over time and is a correlation between the growing number of user and a allegedly decreasing polarity visible?". Figure 4.2 and Figure 4.3 display the average 'compound all' value of each month from August 2008 to November 2019. The average in the C community drops by roughly 0.11 points within five years and slowly starts to ascent since the second half of 2015. The 'compound all' value in the Python community shows a similar steep fall off within the first six years. It lost over 0.15 points and stays around a total of 0.23 points since the second half of 2014.

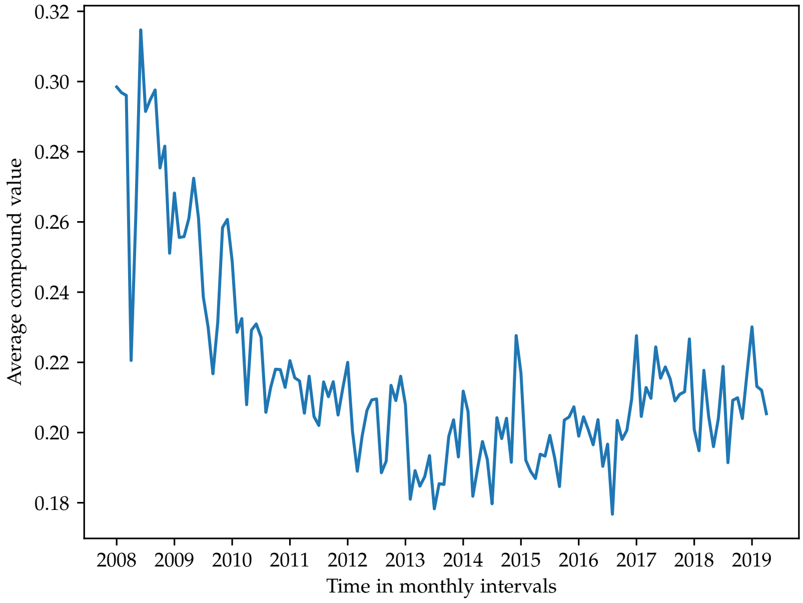

Figure 4.2: Monthly average 'compound all' value in the C community [August 2008, November 2019]

The change of values over time correlate to the number of registered users with a medium strong Pearson correlation of -0.41 for the C community and -0.56 for the Python community. A moderate linear correlation is definitely visible for both communities. These values should not be taken as absolute truth as the number of registered users for the whole of Stack Overflow grew nearly exponentially but only 41.1% of the questioned users in the "Stack Overflow Developer Survey 2019" use Python and only every 5th still uses the C programming language regularly [2]. Additionally does the Pearson correlation not display information whether the change of one value caused the change of the other. However, the rising number of registered users might be one of multiple reasons the overall polarity of answers dropped so significantly within the first years. A quickly growing community might violate community guidelines and rules for questions more often when many of the new users post themselves,

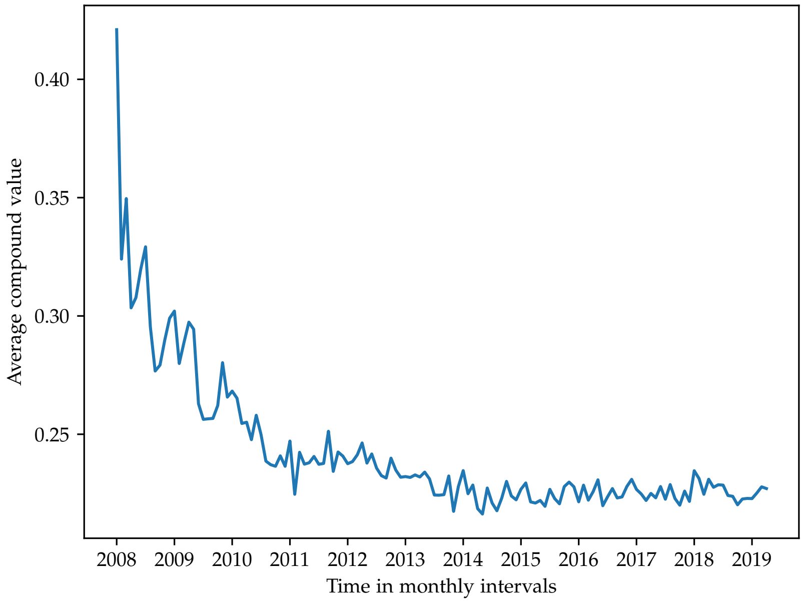

Figure 4.3: Monthly average 'compound all' value in the Python community [August 2008, November 2019]

as shown by Honsel et al. [12]. They showed that users that posted within their first week violate the rules more than twice as often than more experienced users. This can lead to more unfriendly behaviour as a CQ&A forum is built upon these rules to promote higher quality questions and answers. Duplicate questions can annoy experienced users especially quickly because they use their time to read questions that were already answered. More reasons could include unfriendly questions, different rules and perceptions of user interactions as well as different cultural backgrounds and others.

## 4.2 User level questions

### 4.2.1 Do edited answers show exceptional positivity or negativity?

Stack Overflow allows the owner of an answer and users with enough reputation to edit posts. Editing answers can be used to resolve errors within the answer, clarify a statement and extend the level of detail. High reputation users that mastered a certain technology can additionally enhance correct answers with more in-depth knowledge. This may lead to a better understanding for everyone viewing the answer. The editing feature however also allows the owner to reformulate a negative answers in order to make it friendlier and gain more positive feedback through upvotes. Users tend to use these possibilities often as Table 4.3 shows. Over 35% of answers in both communities are edited after the initial posting.

Table 4.3: Number and percentage of edited answers in the C and Python community.

| | C | Python |
|---|---|---|
| Number of edits | 227921 | 708834 |
| Percentage of edited to all answers | 37% | 35% |

With over every third answer edited, the question whether edited answers show different polarity scores than unedited ones arises. In order to classify these results in a better way, the upvotes of an answer are considered as well. The reason is that a low upvote count with a positive polarity value could indicate an originally negative or wrong question, which can also be viewed as negative.

The compound values for both communities show an increase in positivity by 0.03 points for the C and 0.04 points for the Python community (see Table 4.4). Nevertheless removing the edited answers from the rest reveals that the average 'compound all' value of unedited answers is lowered by 0.02 points in both groups. The difference between edited and unedited question is 0.05 points for C and 0.06 for Python, so they are not significant according to the threshold of 0.1 points, but a light lean towards it is visible.

Table 4.4: Mean compound value of the C and Python community for unedited, edited and all answers.

| | C | Python |
|---|---|---|
| Compound of unedited answers | 0.19 | 0.21 |
| Compound of edited answers | 0.24 | 0.27 |
| Compound of all answers | 0.21 | 0.23 |

In order to take a closer look on the edited answers, this paper analyzes other syntactical properties like the comment count and the voting score as well. The comment count displays how many people made a comment on the answer. The voting score indicates whether people found the answer to be helpful for this question, as well how the answer was formulated. A very negative answer might be downvoted by the community even if the content is correct. As stated above, the community uses the feature of editing very often for different possible reasons. The hypotheses that explains the higher compound value is the following: 'Answers that are edited were often improved in polarity and quality. They are given more details compared to answers that were not edited'.

To answer the hypotheses, Table 4.5 and Table 4.6 show relevant syntactical values of unedited and edited answers in both communities respectively. The average values are rounded up, because a floating point voting score or comment count is not possible on Stack

Table 4.5: Rounded up average voting score, comment count and text length as well as the total ranges of voting score and comment count for unedited answers in both communities.

| | C | Python |
|---|---|---|
| Average voting score | 3 | 2 |
| Average comment count | 2 | 1 |
| Average text length (in words) | 70 | 49 |
| Minimum score | -36 | -28 |
| Maximum score | 785 | 1802 |
| Maximum comment count | 72 | 39 |

Table 4.6: Rounded up average voting score, comment count and text length, total ranges of voting scores and comment count of edited answers in both communities.

| | C | Python |
|---|---|---|
| Average voting score | 5 | 6 |
| Average comment count | 3 | 3 |
| Average text length (in words) | 114 | 75 |
| Minimum score | -31 | -56 |
| Maximum score | 3438 | 13974 |
| Maximum comment count | 91 | 64 |

Overflow. As mentioned before, the voting score is a good indication of how helpful and possibly how friendly an answer is. For edited question tagged with C, the average voting score increased by two and tripled for Python. This change in value is significant under the observation that the modal value of voting scores is 0. The reason for this modal value is the absence of votes for every other answer in both communities. The modal value displays the most occurring value for the analyzed attribute in the database. This observation is strengthened by the significant growth of the maximum voting scores. The maximum voting scores increased by over 4.3 and 7.7 times for the communities. At the same time, the minimal voting scores did not change significantly and even shrunk for the C community.

Not only the voting score but also the average number of comments increased for edited answers. For the C community the average answer has 50% more comments than an unedited answer, while a Python answer has three times as many. The increase of one and two comments on average can be viewed as low and not significant. Nevertheless when considering the difference of one and two comments per posts on the scale of Stack Overflow, the edited answers of the C community have over 220.000 additional comments and the edited answers of the Python community additional 1.4 million comments. This is a significant difference of comments between the two categories of answers. Furthermore, the range of comments per answer are 21% and 64% larger for the edited C and Python answers recognizing that the minimal comment count is 0. The higher polarity values combined with the syntactical statistics definitely show a more positive polarity of edited answers compared to unedited answers. With this evidence, the first part of the hypotheses can be accepted.

The second half of the hypotheses could be answered with the length of the texts. Therefore, the words per texts are counted and compared. On average edited answers contain 60% and 53% more words in the C and Python community. However, the increase cannot be used to answer this part of the hypotheses. The reason is the absence of source code in the answers. As described in chapter 2, every piece of code that was marked as code was removed. Code is often used to display examples or direct solutions for a problem. Therefore these word counts cannot represent the actual increase in detail.

At this point it is important to point out that Stack Overflow and its subcommunities display a skewed statistic, because a large amount of the posts does not have any interaction. This results in the modal values which are mostly neutral for each statistical analysis and very low average values.

### 4.2.2 Do the top contributors promote a positive answering culture?

When looking at answers in both communities, the question "Which people contribute the most and how do they answer questions?" emerges. On a website that is built upon a reputation system and allows user with higher reputation to moderate content of other people [13], these experts can serve as a role model and representative of the website. Movshovitz-Attias et al. [14] found users with a high reputation tend to ask and answer more questions, and provide higher quality answers which have a higher probability of becoming the accepted answers. In the Python community, the top five contributors account for 57070 answers, while in the C community the top five users answered 21896 questions in total. The top contributors of both communities started to answer questions in 2008 and 2009 as displayed in Figure 4.4. However, not all of the current most active users started in these years. For both communities a few users started five to six later and started to answer many questions quickly. This effect is described by Movshovitz-Attias et al. [14] who discovered a correlation between the number of answered questions within the first few week of a user's account creation and the later increase of reputation on the website. While this analysis focuses on a broader time scale, the data displays a significant number of answered questions for all ten users within the first few years since the account creation. This supports the original findings. When looking at the years with most answered questions, the Python users posted 8346 answers in 2014 and the C users 3467 in 2015.

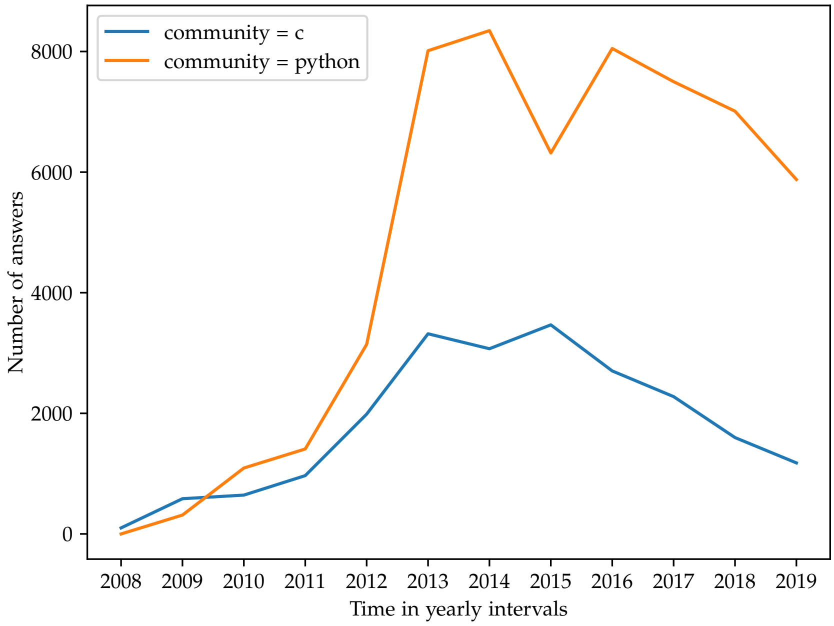

Figure 4.4: Yearly number of answers by the top five contributors of each community [2008-2019]

In order to analyze a possible role model function of users, syntactical values of posts provide one possibility to approach it. However, the average voting score for almost all users declines over time and reaches a stable value range around the average of unedited questions as seen in Table 4.5. Some users had an exceptional voting score on their answers within the

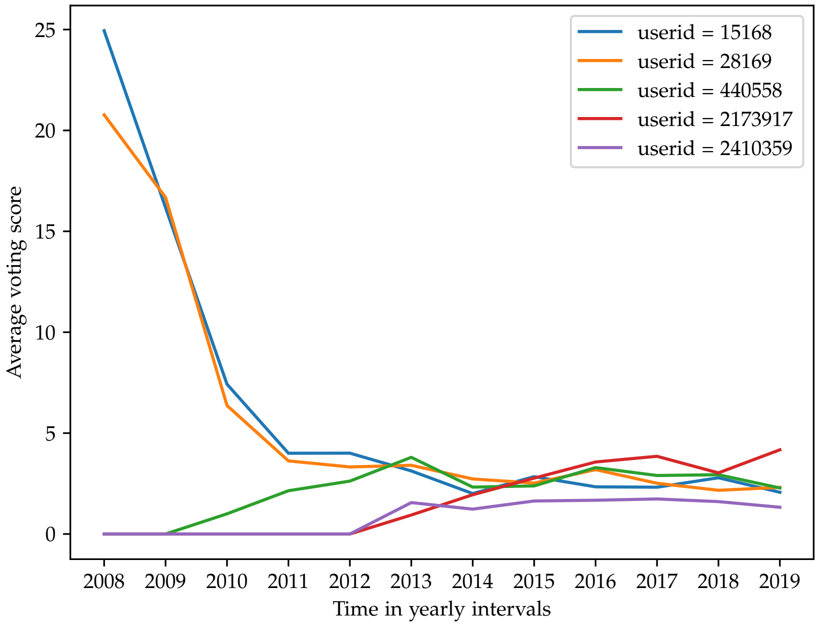

Figure 4.5: Yearly average voting score of the top five C contributors [2009-2019]

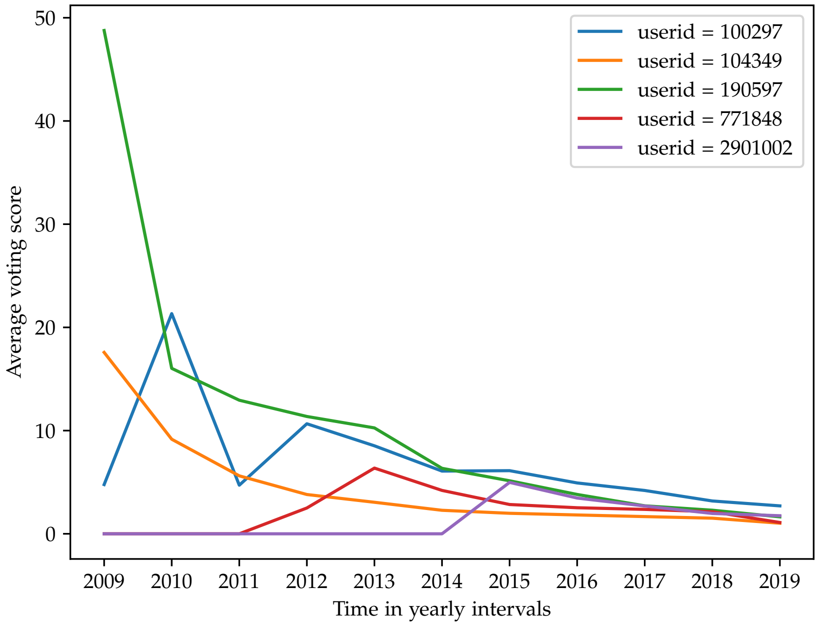

Figure 4.6: Yearly average voting score of the top five Python contributors [2009-2019]

first years, but in these years, the number of contributions was not as high as a few years later. The major reason for this decline in voting score is the number of answers every user writes, because Stack Overflow represents a skewed statistic. The more questions a user answers the more likely he is to gain only a few interactions on it. Additionally the interaction of other users, such as comments or accepting the answer as the solution to the initial question, can increase the visibility and therefore influence the voting score.

In contrast to the converging voting score, the average 'compound all' values of most users tend to rise over time (see Figure 4.7, Figure 4.8). In the beginning of an user's account, the initial compound value tends to decrease for four to five years until it starts to rise steadily. The decreases tend to stop in the years when the number of answers per year reached it's maximum. With a decreasing number of answers, the compound value rises again. The corresponding Pearson correlation of answers count and compound value is -0.12 for the Python community, showing a low linear correlation. The C community has a Pearson correlation of -0.43 indicating a medium linear correlation between the number of answers and the compound value. This correlation does not prove that the number of answers directly influences the polarity of them, especially when taking experience of users into account. The two graphs however indicate that these users gained more experience in answering questions and produce friendlier and more high quality answers while answering less questions within the last years. In order to put the compound values into the context of the whole community, the top five users of the C community do not show a better, but a slightly less positive compound value than the whole of the community in recent years. As shown in Figure 4.2, the community compound average shows a similar curve shape but most of the time at a more positive polarity. Until the decline of polarity for the Python community stopped, the top contributors show a similar decline with slightly less positivity than the community average. In contrast to the stagnating community compound, the polarity of four top users rises above the community level.

These results indicate two different answers for the hypotheses. For the C community, the top five users do not show special properties compared to the community average. Their polarity is even slightly less than the average but still very positive. However, the Python users clearly show more positive answers, while staying close to the community average within the first years. Especially in recent years the polarity starts to increase over the community average. While the top C contributors cannot be classified as a role model with exceptional positive and good answers, the best Python contributors are slowly growing into this position.

## 4.3 The quality of VADER for Software Engineering texts

In order to analyze the actual results of the VADER sentiment analysis tool, questions of different polarity ranges are chosen randomly and manually scored. This random sampling in the value ranges is used to counter act against choosing very well fitting polarity scores. The random numbers are drawn with a Python script that takes the current time as seed

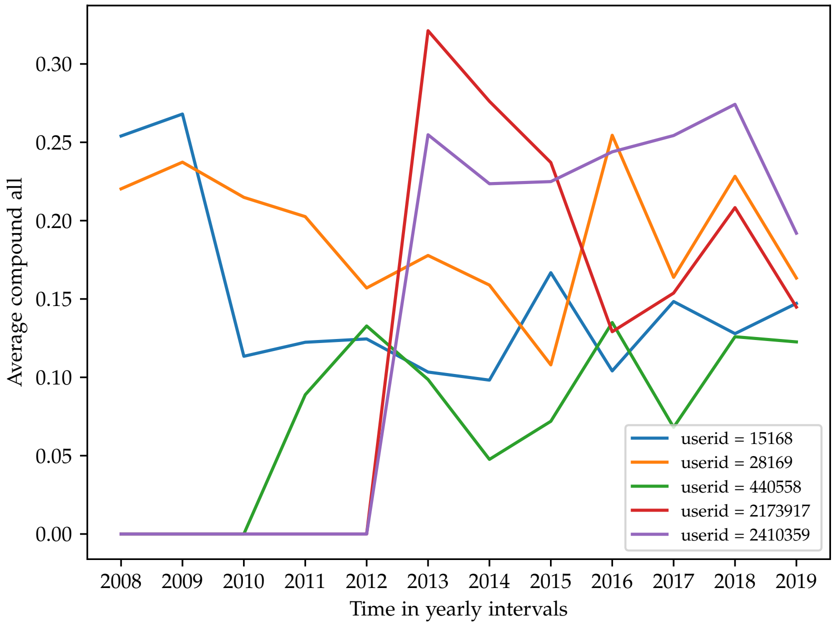

Figure 4.7: Yearly average ′compound all′ value of the top five C contributors [2008-2019]

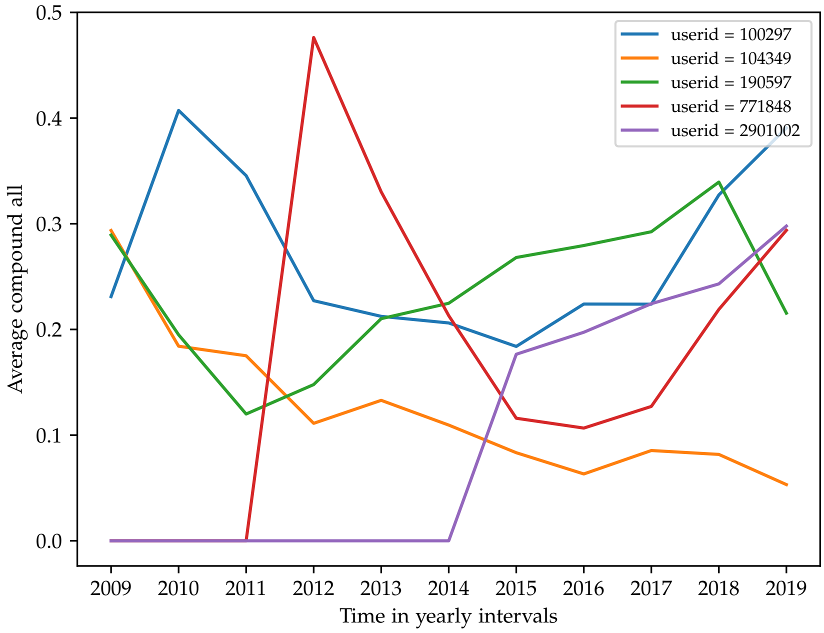

Figure 4.8: Yearly average ′compound all′ value of the top five Python contributors [2009-2019]

for the pseudo number generator. The chosen numbers are the row number of the chosen tuples, which are ordered by ascending post id to allow consistent results from the database. Afterwards, the differences of the two compound values are analyzed and discussed for the example questions examined before.

To gain a better understanding of the reasoning behind VADER, this paper sets six value ranges for the polarity of posts. These ranges can be seen in Table 4.7. The interval borders between very negative and negative as well as very positive and positive are set to split the two large intervals into four distinct ones to gain a better insight into the very positive and very negative answers. The neutral interval is set according to the initial definition in subsection 4.1.1. The polarity of exactly 0 is examined separately from the neutral posts because 93.3% of the neutral posts have a 0 'compound all' value and it was very unlikely to sample more than two out of 20 neutral posts that did not have 0 as compound value. This would complicate the examination of other neutral values. To influence the manual rating as little as possible, a script which only displays the text that has to be rated with no other information was used. In the rating results, a value disparity greater than 0.1 between the manual rating and the VADER rating is classified as significant. This threshold corresponds to the size of the neutral interval. For the analysis the average voting score on the 20 answers per category were calculated as well as the number of ratings which were within the 0.1 polarity range for the same answer. The results of the manual rating show that most of the time the manual rating and automated rating are significantly different. The rating of answers with very positive and very negative score does not match with the others as seen in Table 4.8. Very positive manual ratings are not existent in the sample because no question reached a value higher than 0.6, while the interval only starts at 0.75 points. The very negative answers also show a high value difference, because the manual rating is on average neutral and not negative or very negative. Negative answers are rated as neutral in the manual rating, compared to the VADER rating which lies in the center of the negative interval. Only one answer is rated similarly in this category. When looking at the neutral interval, nearly 50% of the answers are rated similarly. The average difference is above the 0.1 threshold and is therefore still significant. For answers that have a compound value of 0, all answers are rated the same and the average is also equivalent. In both ratings, the positive answers are rated

Table 4.7: The compound value intervals with associated polarity.

| Value intervals | associated polarity |
|---|---|
| [-1.0, -0.75) | very negative |
| [-0.75, -0.05) | negative |
| [-0.05, 0) and (0, 0.05] | neutral |
| [0, 0] | exactly neutral |
| (0.05, 0.75] | positive |
| (0.75, 1.0] | very positive |

Table 4.8: Average polarity value of the manual rating and the VADER rating and number of similar ratings.

| Polarity | Manual rating | VADER rating | number of similar scores |
|---|---|---|---|
| very negative | -0.032 | -0.866 | 0 |
| negative | -0.003 | -0.436 | 1 |
| neutral | 0.125 | 0.005 | 9 |
| exactly neutral | 0 | 0 | 20 |
| positive | 0.063 | 0.415 | 2 |
| very positive | 0.270 | 0.894 | 0 |

as positive and two answers are rated similarly. However, the two average values have a discrepancy greater than 0.3 and are therefore not rated equally.

These results can be explained by looking at the answers that were rated [15]. Very negative answers usually contain questions from the users to the question owner or the answering person structured the answer with questions that are then answered. Additionally they can contain texts from documentations, which are not rated as positive. These answers also contain many software engineering related abbreviations which can influence the rating negatively as they might not fit to any of the dictionary entries of VADER. Another factor is the terminology of Software Engineering, where phrase like "termination" is associated with processes and code execution. This term however does not have a positive connotation for normal conversations. Other phrases that can have a negative tone include "execute", "killing" or "timeout" which are also often used with processes. The opposite site of compound values contains very positive answers. These were classified as positive in the manual rating. However, their average score does not reach the automated level because two answers were rated as negative because of the way the sentences are built. These answers contained short and harsh sentences that directly talked to the question owner and state that the usual solution would be better than what the question's solution was. However, most of the very positive questions are positive or neutral. Their vocabulary is friendlier and does not address the questioner in a direct and harsh way. The main difference from the VADER rating comes from a subjective view on what a very positive question is, and how it should look like. The two categories of positive and negative answers fall in between the two extremes, where the discussed aspects such as Software Engineering specific vocabulary, abbreviations influence the automated rating. The two neutral ratings show the best performance when looking at Table 4.8. The category of exactly 0 polarity is a perfect fit, because all questions from this category contain very few sentences which are originally pointing to some sort of source code or link. Because these two text types are removed in the data set, these answers cannot be rated properly. The answers that fall into the neutral category mostly contain simple answers with a short explanation or no complete sentence at all. Additionally a short paragraph from a documentation may be found, which shifts the whole answer into a objective view.

Therefore these answers are rated very similarly with only a few exceptions.

After analyzing the results of the manual and automated rating, the question "How does the 'compound average' value perform?" comes up. The 'compound average' value is calculated by inputting single lines of the answer into the sentiment analyzer and calculating the average overall results. In contrast the 'compound all' score is calculated by inputting the whole answer body into the sentiment analyzer. The 'compound all' value is used for this paper, because from my understanding of the tool, different phrases and sentence constructs can be specifically evaluated. Therefore the split of answers into multiple lines may lead to the split of a coherent sentence and therefore alter the result. This problem is especially present for long answers with many paragraphs. These effects can be seen when in the database, which contains over 1.6 million answers which have a different 'compound all' and 'compound average' value. However, when looking at Table 4.9 together with Table 4.8, it is visible that the 'compound average' value is closer to the manual rating than the 'compound all' value. The average of the very positive answers would be in the 0.1 range of the manual rating and therefore considered similar. Furthermore the values of the very positive and very negative answers moved into the category of positive and negative answers. Therefore the two boundary categories disappear which would make a new drawing of 120 answers necessary to reflect the range of possible values. This behaviour of VADER is not very stable as mentioned before.

Table 4.9: The average 'compound average' and 'compound all' value of VADER for the 120 questions.

| Polarity | VADER compound avg | VADER compound all |
|---|---|---|
| very negative | -0.231 | -0.866 |
| negative | -0.206 | -0.436 |
| neutral | -0.008 | 0.005 |
| exactly neutral | 0 | 0 |
| positive | 0.190 | 0.415 |
| very positive | 0.306 | 0.894 |

# 5 Discussion

The results presented in chapter 4 reveal three main conclusions. The first conclusion is that the C and Python community can be considered positive and relatively friendly when analyzing the answers of users with automated tools. However the polarity of both communities decline over time while Stack Overflow is growing stronger than before. While the growing number is not the only reason for this decline, it is definitely part of it, especially when considering the results of Honsel et al. [12] and the rise of external as well as internal opinions on the website. The second important result of this paper is the way features can be used to influence the polarity on the website. The editing feature leads to more interaction, more detailed and friendlier questions. In the Python community, edited questions reached on average higher voting scores, comment counts but also lower voting scores when looking at the range of possible scores. The C community shows similar but weaker behaviour and tends to promote edited questions. The reputation feature, which was added to give experienced users moderation rights as well as more visibility through extra badges, does not lead to exceptional positive and friendly answers. The top five users of the C and Python community stay close but below the average of the community, which they also help to create with the amount of answers they generate. However in recent years, four of the top five Python contributors increased their polarity above the community average. This increase in polarity indicates that they gain more experience over the years and are able to produce positive and friendly answers within their community, in the end representing them.

The last major result is the way the sentiment analysis tool VADER behaves for Software Engineering texts. The different behaviour of VADER for different inputs reveals a definite weakness of the tool. These differences are especially important considering that they can be larger than the recommended neutral interval. This can lead to two different interpretations of the same text. After presenting the data for the average compound values, which are much closer to the manual rating than the all compound values, it is still open for further research whether this tool can be used for Software Engineering texts. It is also important to note, that only 120 posts were rated manually so these effects can change when rating a larger set of answers. However with the sampling process the best way to gain a representative sample was used. At the current state, the polarity values alone are not enough to classify answers of the CQ&A website Stack Overflow in a meaningful way and additional indices such as the voting score, comment count or other interactions should be added to the analysis.

# 6 Outlook

While the sentiment analysis tool VADER can be analyzed in further research, it is also important to broaden the horizon and include other tools. One such tool is Senti4SD that was specifically attuned towards Software Engineering texts and uses a different approach than VADER. It is a classifier which uses "a suite of both lexicon- and keyword-based features, as well as semantic features based on word embedding" [16]. Other possible tools are 'Standord Core NLP', 'SentiStrength' and others described and analyzed for Stack Overflow texts by Lin et al [17] during their research on pattern based mining on Q&A websites.

Additional research fields on Stack Overflow are younger communities such as Smart Contracts or the Rust programming language. These communities were created only a few years ago because the technology did not exist before. It might be interesting to see how users of older technologies such as C, Java, C++ and others migrated over time and took their answering culture to these communities. Especially on the background of Figure 2.1 which clearly shows a steady decrease of answers in the C community indicating that this community will only receive answers from very few remaining developers.

In the end the challenge to build a classifier which has a high accuracy and can be used by Stack Overflow to improve the website remains.

# List of Figures

# List of Tables

# Bibliography

[1] David Fullerton, *State of the Stack 2019*. [Online]. Available: https://stackoverflow.blog/2019/01/18/state-of-the-stack-2019-a-year-in-review/ (visited on 01/02/2020).

[2] StackOverflow, *Developer Survey Results 2019*, 2019. [Online]. Available: https://insights.stackoverflow.com/survey/2019?utm%7B%5C_%7Dsource=Iterable%7B%5C&%7Dutm%7B%5C_%7Dmedium=email%7B%5C&%7Dutm%7B%5C_%7Dcampaign=dev-survey-2019 (visited on 02/01/2020).

[3] J. Hanlon, *Stack Overflow Isn't Very Welcoming. It's Time for That to Change.* 2018. [Online]. Available: https://stackoverflow.blog/2018/04/26/stack-overflow-isnt-very-welcoming-its-time-for-that-to-change/ (visited on 01/31/2020).

[4] J. Silge and J. Punyon, *Classifying-comments-on-stack-overflow*, 2018. [Online]. Available: https://stackoverflow.blog/2018/07/10/welcome-wagon-classifying-comments-on-stack-overflow/.

[5] C. J. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text", in *Eighth international AAAI conference on weblogs and social media*, 2014.

[6] Stack Exchange Inc, *Archive.org-StackExchange*. [Online]. Available: https://archive.org/download/stackexchange/ (visited on 01/03/2020).

[7] StackExchange, *StackExchange-DataExplorer*. [Online]. Available: https://data.stackexchange.com/stackoverflow/query/new (visited on 01/03/2020).

[8] L. Hohmann, *Code Repository*, 2019. [Online]. Available: https://github.com/GilgusMaximus/PolarityMining-StackOverflow.

[9] Shicks, *Stackoverflow Is A Difficult Community to Participate In*, 2012. [Online]. Available: https://theexceptioncatcher.com/blog/2012/09/stackoverflow-is-a-difficult-community-to-participate-in/%7B%5C#%7D.UEu2l1DNwcU.hackernews (visited on 01/31/2020).

[10] J. Slegers, *The decline of Stack Overflow*, 2015. [Online]. Available: https://hackernoon.com/the-decline-of-stack-overflow-7cb69faa575d (visited on 01/31/2020).

[11] B. Leggiero, *Why does SE appear a rather unfriendly environment for new users, and how can we work on fixing that?*, 2015. [Online]. Available: https://meta.stackexchange.com/questions/242490/why-does-se-appear-a-rather-unfriendly-environment-for-new-users-and-how-can-we (visited on 01/31/2020).

[12] V. Honsel, S. Herbold, and J. Grabowski, “Intuition vs. truth: Evaluation of common myths about StackOverflow posts”, *IEEE International Working Conference on Mining Software Repositories*, vol. 2015-Augus, pp. 438–441, 2015, ISSN: 21601860. DOI: `10.1109/MSR.2015.58`.

[13] *Stack Overflow Privileges*. [Online]. Available: `https://stackoverflow.com/help/privileges` (visited on 02/07/2020).

[14] D. Movshovitz-Attias, Y. Movshovitz-Attias, P. Steenkiste, and C. Faloutsos, “Analysis of the reputation system and user contributions on a question answering website: StackOverflow”, *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2013*, pp. 886–893, 2013. DOI: `10.1145/2492517.2500242`.

[15] L. Hohmann, *Manually rated answers*, 2020. [Online]. Available: `https://github.com/GilgusMaximus/PolarityMining-StackOverflow/blob/master/DataAnalysis/manually_rated_answers.txt`.

[16] F. Calefato, F. Lanubile, F. Maiorano, and N. Novielli, “Sentiment Polarity Detection for Software Development”, *Empirical Software Engineering*, vol. 23, no. 3, pp. 1352–1382, Jun. 2018, ISSN: 1573-7616. DOI: `10.1007/s10664-017-9546-9`. [Online]. Available: `https://doi.org/10.1007/s10664-017-9546-9`.

[17] B. Lin, F. Zampetti, G. Bavota, M. Di Penta, and M. Lanza, “Pattern-Based Mining of Opinions in Q&A Websites”, *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pp. 548–559, 2019, ISSN: 02705257. DOI: `10.1109/icse.2019.00066`.